OPOS Services

User Guide

symbol®
The Enterprise Mobility Company™

OPOS Services

User Guide

72E-86287-01

Revision A

November 2006

symbol®

The Enterprise Mobility Company™

© 2006 by Symbol Technologies, Inc. All rights reserved.

No part of this publication may be reproduced or used in any form, or by any electrical or mechanical means, without permission in writing from Symbol. This includes electronic or mechanical means, such as photocopying, recording, or information storage and retrieval systems. The material in this manual is subject to change without notice.

The software is provided strictly on an “as is” basis. All software, including firmware, furnished to the user is on a licensed basis. Symbol grants to the user a non-transferable and non-exclusive license to use each software or firmware program delivered hereunder (licensed program). Except as noted below, such license may not be assigned, sublicensed, or otherwise transferred by the user without prior written consent of Symbol. No right to copy a licensed program in whole or in part is granted, except as permitted under copyright law. The user shall not modify, merge, or incorporate any form or portion of a licensed program with other program material, create a derivative work from a licensed program, or use a licensed program in a network without written permission from Symbol. The user agrees to maintain Symbol’s copyright notice on the licensed programs delivered hereunder, and to include the same on any authorized copies it makes, in whole or in part. The user agrees not to decompile, disassemble, decode, or reverse engineer any licensed program delivered to the user or any portion thereof.

Symbol reserves the right to make changes to any software or product to improve reliability, function, or design.

Symbol does not assume any product liability arising out of, or in connection with, the application or use of any product, circuit, or application described herein.

No license is granted, either expressly or by implication, estoppel, or otherwise under any Symbol Technologies, Inc., intellectual property rights. An implied license only exists for equipment, circuits, and subsystems contained in Symbol products.

Symbol, Spectrum One, and Spectrum24 are registered trademarks of Symbol Technologies, Inc. Bluetooth is a registered trademark of Bluetooth SIG. Microsoft, Windows and ActiveSync are either registered trademarks or trademarks of Microsoft Corporation. Other product names mentioned in this manual may be trademarks or registered trademarks of their respective companies and are hereby acknowledged.

Symbol Technologies, Inc.
One Symbol Plaza
Holtsville, New York 11742-1300
http://www.symbol.com

Revision History

Changes to the original manual are listed below:

Change	Date	Description
-01 Rev A	11/2006	Initial Release. (Hypercom Rev. C)

Table of Contents

Revision History iii

About This Guide
- Introduction iii
- Chapter Descriptions iv
- Notational Conventions iv
- Related Documents v
- Service Information v
  - Global Customer Interaction Center v

Chapter 1: Getting Started
- Introduction 1-1
- OLE for Retail POS 1-1
- General OPOS Sequence 1-1
- Installing OPOS 1-2
- Starting OPOS 1-2

Chapter 2: Implementing Device Control
- Introduction 2-1
- Implementing Signature Capture 2-2
- Implementing MSR 2-5
- Implementing PIN Pad 2-15
- Implementing Line Display 2-23
- Implementing POS Keyboard 2-35

Chapter 3: Adding Direct IO Service Information
- Introduction 3-1

Chapter 4: Using Event Viewer
- Introduction 4-1
- Starting Event Viewer 4-1
- Event Information 4-2
- Examples of Severity Levels 4-3
- Configuring Logging Levels 4-3
- Registry Key Name 4-3

About This Guide

Introduction

This guide provides information about the following features and procedures:

- Installing and opening Symbol OPOS Services.
- Implementing Signature Capture, MSR (Magnetic Stripe Reader), PIN Pad, Line Display and POS Keyboard.
- Providing events to the application that are not otherwise supported by Device Control.
- Event Viewer.

This guide is intended for network administrators, merchants, operators, technicians or those who oversee the configuration and daily maintenance of the terminals and those who load terminal applications and packing lists.

This guide assumes that the user is familiar with the following:

- General characteristics of POS peripheral devices
- Characteristics of the specific peripheral devices to be supported
- OPOS terminology and architecture
- OPOS models.

IMPORTANT This guide includes information about the Symbol PD8700 and PD4700 series terminals, Symbol OEM products from Hypercom Corporation.

Any references in this guide to Hypercom Corporation, Hypercom logo, Hypercom file names and file paths, Hypercom software and terminals reflect hardware and software manufactured by Hypercom Corporation for Symbol Technologies, Inc.

NOTE Screens and windows pictured in this guide are samples and can differ from actual screens.

Chapter Descriptions

Topics covered in this guide are as follows:

- Chapter 1, Getting Started describes how to install and open Symbol OPOS Services.
- Chapter 2, Implementing Device Control includes procedures to implement each of the following device categories: Signature Capture, Magnetic Stripe Reader (MSR), PIN Pad, Line Display and POS Keyboard.
- Chapter 3, Adding Direct IO Service Information includes a procedure to provide events to the application that are not otherwise supported by Device Control.
- Chapter 4, Using Event Viewer describes the Event Viewer feature, explains logging levels and event information, and provides examples of severity levels and logged messages

Notational Conventions

The following conventions are used in this document:

- The term “POS Designer” in this guide refers to software.
- *Italics* are used to highlight the following:
  - Chapters and sections in this and related documents
  - Drop-down list and list box names
  - Check box and radio button names
  - Icons on a screen.
- **Bold** text is used to highlight the following:
  - Names of windows
  - Dialog box components.
- bullets (•) indicate:
  - Action items
  - Lists of alternatives
  - Lists of required steps that are not necessarily sequential
- Sequential lists (e.g., those that describe step-by-step procedures) appear as numbered lists.
- Special icons:

NOTE Notes contain neutral or positive information supplementing the main text. It is often information that applies only to special cases.

IMPORTANT Important statements draw attention to information crucial to using the product successfully. Pay special attention to Important statements.

CAUTION Cautions advise that a negative result, such as a loss of data, may occur.

WARNING! **Warnings provide information that is essential to the safety of the user, the equipment, or both. Failure to do as instructed may result in physical damage.**

Related Documents

For the latest version of this and all payment solutions guides, go to:http://www.symbol.com/manuals.

Service Information

For service information, warranty information, technical assistance or problems with the equipment, contact the regional Symbol Global Customer Interaction Center. Before calling, have the model number, serial number and several bar code symbols at hand.

Call the Global Customer Interaction Center from a phone near the scanning equipment so that the service person can try to troubleshoot the problem. If the equipment is found to be working properly and the problem is reading bar codes, the Support Center will request samples of the bar codes for analysis at our plant.

If the problem cannot be solved over the phone, it may be necessary to return the equipment for servicing. If that is necessary, the Global Customer Interaction Center will provide specific directions.

NOTE Symbol Technologies is not responsible for any damages incurred during shipment if the approved shipping container is not used. Shipping the units improperly can possibly void the warranty. If the original shipping container was not kept, contact Symbol to have another sent.

If the Symbol product was purchased from a Symbol Business Partner, contact that Business Partner for service.

Global Customer Interaction Center

The addresses and telephone numbers below are provided for you convenience. However, this information can change due to telephone provider updates. For the most up-to-date contact number information, visit: www.symbol.com/contactsupport for a Customer Interaction Center in your area.

Country/Region	Address	Telephone
United States	Symbol Technologies, Inc. One Symbol Plaza Holtsville, New York 11742-1300	1-800-653-5350
Canada	Symbol Technologies Canada, Inc. 5180 Orbitor Drive Mississauga, Ontario, Canada L4W 5L9	1-866-416-8545 (Inside Canada) 905-629-7226 (Outside Canada)
United Kingdom	Symbol Technologies Symbol Place Winnersh Triangle, Berkshire RG41 5TP United Kingdom	0800 328 2424 (Inside UK) +44 118 945 7529 (Outside UK)

Country/Region	Address	Telephone
Asia/Pacific	Symbol Technologies Asia, Inc. (Singapore Branch) 230 Victoria Street #12-06/10 Bugis Junction Office Tower Singapore 188024	Tel: +65-6796-9600 Fax: +65-6337-6488
Australia	Symbol Technologies Pty. Ltd. 432 St. Kilda Road Melbourne, Victoria 3004	1-800-672-906 (Inside Australia) +61-3-9866-6044 (Outside Australia)
Österreich/Austria	Symbol Technologies Austria GmbH Prinz-Eugen Strasse 70 / 2.Haus 1040 Vienna, Austria 01-5055794-0 (Inside Austria)	+43-1-5055794-0 (Outside Austria)
Danmark/Denmark	Symbol Technologies AS Dr. Neergaardsvej 3 2970 Hørsholm	7020-1718 (Inside Denmark) +45-7020-1718 (Outside Denmark)
Europe/Mid-East Distributor Operations		Contact your local distributor or call +44 118 945 7360
Suomi/Finland	Oy Symbol Technologies Kaupintie 8 A 6 FIN-00440 Helsinki, Finland	9 5407 580 (Inside Finland) +358 9 5407 580 (Outside Finland)
France	Symbol Technologies France Centre d'Affaire d'Antony 3 Rue de la Renaissance 92184 Antony Cedex, France	01-40-96-52-21 (Inside France) +33-1-40-96-52-50 (Outside France)
Deutschland/ Germany	Symbol Technologies GmbH Waldstrasse 66 D-63128 Dietzenbach, Germany	6074-49020 (Inside Germany) +49-6074-49020 (Outside Germany)
Italia/Italy	Symbol Technologies Italia S.R.L. Via Cristoforo Columbo, 49 20090 Trezzano S/N Navigilo Milano, Italy	2-484441 (Inside Italy) +39-02-484441 (Outside Italy)
Latin America Sales Support	2730 University Dr. Coral Springs, FL 33065 USA	1-800-347-0178 (Inside United States) +1-954-255-2610 (Outside United States) 954-340-9454 (Fax)

Country/Region	Address	Telephone
México/Mexico	Symbol Technologies Mexico Ltd. Torre Picasso Boulevard Manuel Avila Camacho No 88 Lomas de Chapultepec CP 11000 Mexico City, DF, Mexico	5-520-1835 (Inside Mexico) +52-5-520-1835 (Outside Mexico)
Nederland/ Netherlands	Symbol Technologies Kerkplein 2, 7051 CX Postbus 24 7050 AA Varsseveld, Netherlands	315-271700 (Inside Netherlands) +31-315-271700 (Outside Netherlands)
Norge/Norway	Symbol's registered and mailing address: Symbol Technologies Norway Hoybratenveien 35 C N-1055 OSLO, Norway Symbol's repair depot and shipping address: Symbol Technologies Norway Enebakkveien 123 N-0680 OSLO, Norway	+47 2232 4375
South Africa	Symbol Technologies Africa Inc. Block B2 Rutherford Estate 1 Scott Street Waverly 2090 Johannesburg Republic of South Africa	11-809 5311 (Inside South Africa) +27-11-809 5311 (Outside South Africa)
España/Spain	Symbol Technologies S.L. Avenida de Bruselas, 22 Edificio Sauce Alcobendas, Madrid 28108 Spain	91 324 40 00 (Inside Spain) +34 91 324 40 00 (Outside Spain) Fax: +34.91.324.4010
Sverige/Sweden	"Letter" address: Symbol Technologies AB Box 1354 S-171 26 SOLNA Sweden Visit/shipping address: Symbol Technologies AB Solna Strandväg 78 S-171 54 SOLNA Sweden	Switchboard: 08 445 29 00 (domestic) Call Center: +46 8 445 29 29 (international) Support E-Mail: Sweden.Support@se.symbol.com

Chapter 1 Getting Started

Introduction

This chapter includes procedures for installing and starting Symbol OPOS Services.

OLE for Retail POS

OLE for Retail POS (OPOS) was the first widely-adopted POS device standard to help integrate POS hardware into Windows-based applications. Symbol OPOS drivers are plug-and-play drivers that provide support for MSR, PIN Pad, Signature Capture, POS Keyboard, and Line Display. OPOS uses COM technology, and is therefore language independent.

OPOS consists of:

- An architecture for Win32-based POS device access
- A set of POS device interfaces sufficient to support a wide range of POS solutions

In the general OPOS model, OPOS controls adhere to the ActiveX control specifications. They expose properties, methods, and events to a containing application. The controls are invisible at run time, and rely exclusively on the application for requests through methods and sometimes properties. Responses are given to the application through method return values and parameters, properties, and events.

General OPOS Sequence

Before you open Symbol OPOS Services and begin to configure your devices, here is a quick review of the sequence of actions used by OPOS. In general, OPOS follows this sequence to open, use, and close a Device:

1. Obtain a **Device Control** reference to select the name of the Device Service.

   NOTE For Symbol OPOS Services, the Device Service may be MSR, PIN Pad, Signature Capture, POS Keyboard, or Line Display.

2. Call the **Open** method to start a Device Service and link it to the Device Control.
3. Call the **Claim** method to gain access to the Physical Device.
4. Set the **DeviceEnabled** property to true to make the Physical Device operational.

5. For shareable devices, the Device may be enabled without first claiming it.
6. Set the **DataEventEnabled** property to true to deliver the received data as a Data Event.
7. Call the **Release** method to release access to the Physical Device.
8. Call the **Close** method to unlink the Device Service from the Device Control.

Installing OPOS

The Symbol OPOS Services application is included on the FPE32 Developer's Toolkit CD. The CD includes an autorun function that prompts you to install the application to your hard drive. Follow the on-screen steps to install OPOS.

Starting OPOS

To start the OPOS application:

1. Double click the OPOS icon (*Figure 1-1*) that was installed onto your desktop.

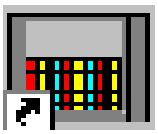

Figure 1-1 *OPOS Icon*

A window similar to the following OPOS window displays.

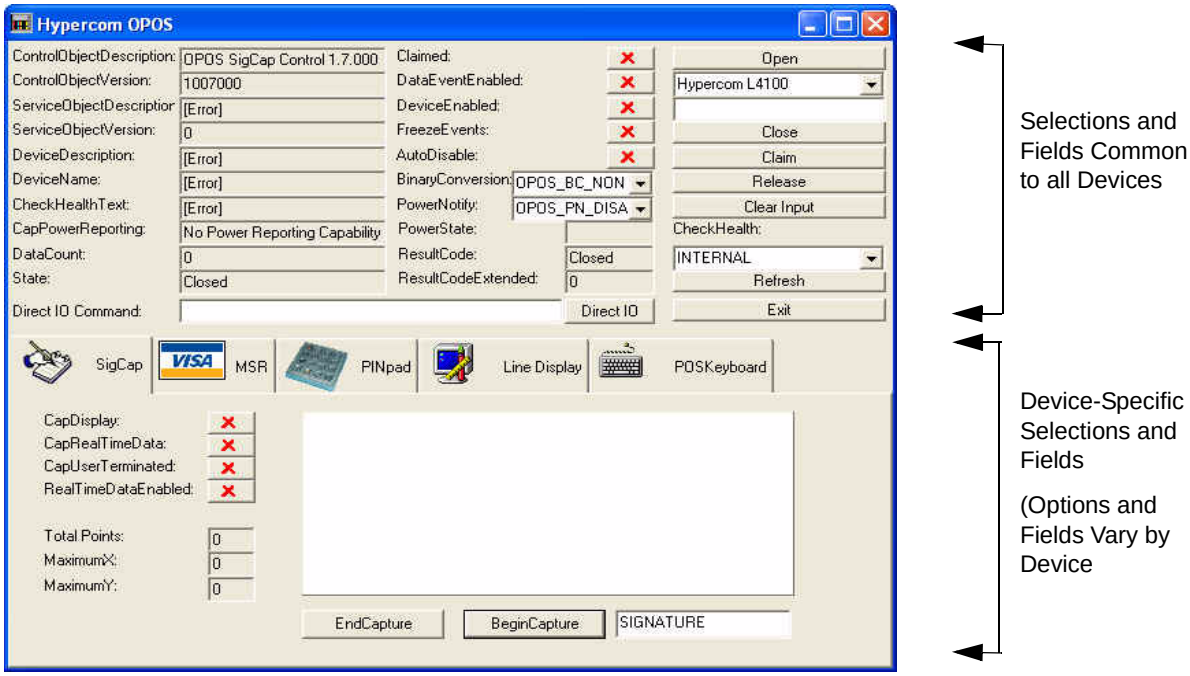

Figure 1-2 *OPOS Main Window*

The OPOS main window has two main parts:

- The upper portion of the OPOS window contains selections common to all supported devices.
- The lower portion of the window changes depending on which device you select from the tabs and contains device-specific selections.

2. Continue with Chapter 2, Implementing Device Control.

Chapter 2

Implementing Device Control

Introduction

This chapter includes procedures to implement each of the following device categories:

- Signature Capture
- MSR
- PIN Pad
- Line Display
- POS Keyboard

Implementing Signature Capture

The Signature Capture Device captures signature data. The captured signature data is in the form of lines consisting of a series of points. Each point lies within the coordinate system defined by the resolution of the device, where (0,0) is the upper-left point of the device, and (MaximumX, MaximumY) is the lower-right point. The signature line points are presented to the application by a DataEvent with a single array of line points.

If enabled by the application, the Signature Capture Device can also:

- Provide a way for the user to terminate signature capture.
- Display data on the signature capture device.
- Return the signature in “real time” as it is entered on the device.

The Sig Cap tab on the OPOS window allows you to open and use the Signature Capture Device Service. See the following example of an OPOS window with the Sig Cap tab selected.

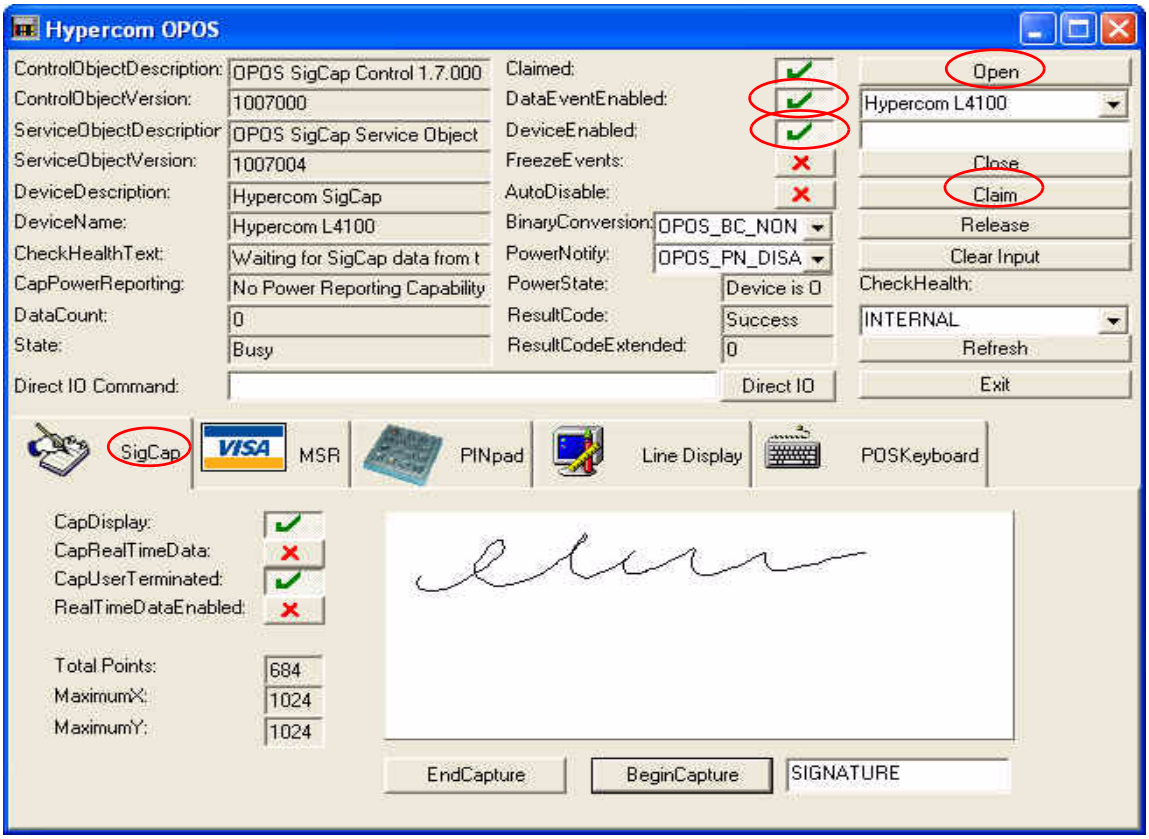

Figure 2-1 *OPOS Signature Capture Tab*

To configure Signature Capture:

1. From the OPOS screen, select the name of the terminal from the Terminal pull-down menu or ensure that the correct terminal is selected in the upper right corner of the screen.
2. Click the **Sig Cap** tab.
3. Click **Open**.
4. Click **Claim**.

   NOTE When the Claim button is clicked, the **`Claimed`**: field at the top of the OPOS window changes from a red "X" to a green checkmark.

5. Click **Device Enabled**.

   NOTE When DeviceEnabled is selected, the red "X" on the button changes to a green checkmark.

6. Click **DataEvent Enabled**.

   NOTE When DataEventEnabled is selected, the red "X" on the button changes to a green checkmark.

7. To capture a signature, verify the signature form name and click **beginCapture**.

   NOTE If the CapDisplay property is true, then formName is used to find information about the form or data screen to be displayed. After displaying the form or data screen, when applicable, the signature capture stylus is enabled.

8. To stop capturing a signature, click **endCapture**.

   NOTE If RealTimeDataEnabled is false and a signature was captured, then it is placed in the properties PointArray and RawData. If no signature was captured, then PointArray and RawData are set to a length of zero. If RealTimeDataEnabled is true and there are signature points remaining which have not been delivered to the application by a DataEvent, then the remaining signature is placed into the properties PointArray and RawData. If no signature was captured or all signature points have been delivered to the application, then PointArray and RawData are set to a length of zero.

9. If you need to view attributes listed on the OPOS Signature Capture window, see *Table 2-1* for descriptions of the properties.

Table 2-1 *Signature Capture Property Descriptions*

Property	Description	How Initialized
CapDisplay	If true, the device is able to display a form or data entry screen.	Open method.
CapRealTimeData	If true, the device is able to supply signature data as the signature is being captured ("real time").	Open method.
CapUserTerminated	If true, the user is able to terminate signature capture by checking a completion box, pressing a completion button, or performing some other interaction with the device. If false, the application must end signature capture by calling the endCapture method.	Open method.

Table 2-1 *Signature Capture Property Descriptions (Continued)*

Property	Description	How Initialized
CapDisplay	If true, the device is able to display a form or data entry screen.	Open method.
CapRealTimeData	If true, the device is able to supply signature data as the signature is being captured (“real time”).	Open method.
CapUserTerminated	If true, the user is able to terminate signature capture by checking a completion box, pressing a completion button, or performing some other interaction with the device. If false, the application must end signature capture by calling the endCapture method.	Open method.
DeviceEnabled	If true, the signature capture device is enabled. If CapDisplay is true, then the display screen of the device is cleared.	Initialized to false by the open method.

Implementing MSR

The MSR (Magnetic Stripe Reader) Device supports the attachment of a card reader to provide input to the application from a swiped or waved card. The targeted environment is electronic funds data, such as an account number or a customer name, from magnetically encoded credit and/or debit card.

The MSR Device can:

- Read encoded data from a magnetic stripe or waved card.
- Support decoding of alphanumeric data bytes into their corresponding alphanumeric codes.

MSR may also support the following additional capabilities:

- Specific card types: ISO, JIS Type I.
- Returning the track sentinels with track data.

The MSR tab on the OPOS window allows you to open and use MSR. See the following example of an OPOS window with the MSR tab selected:

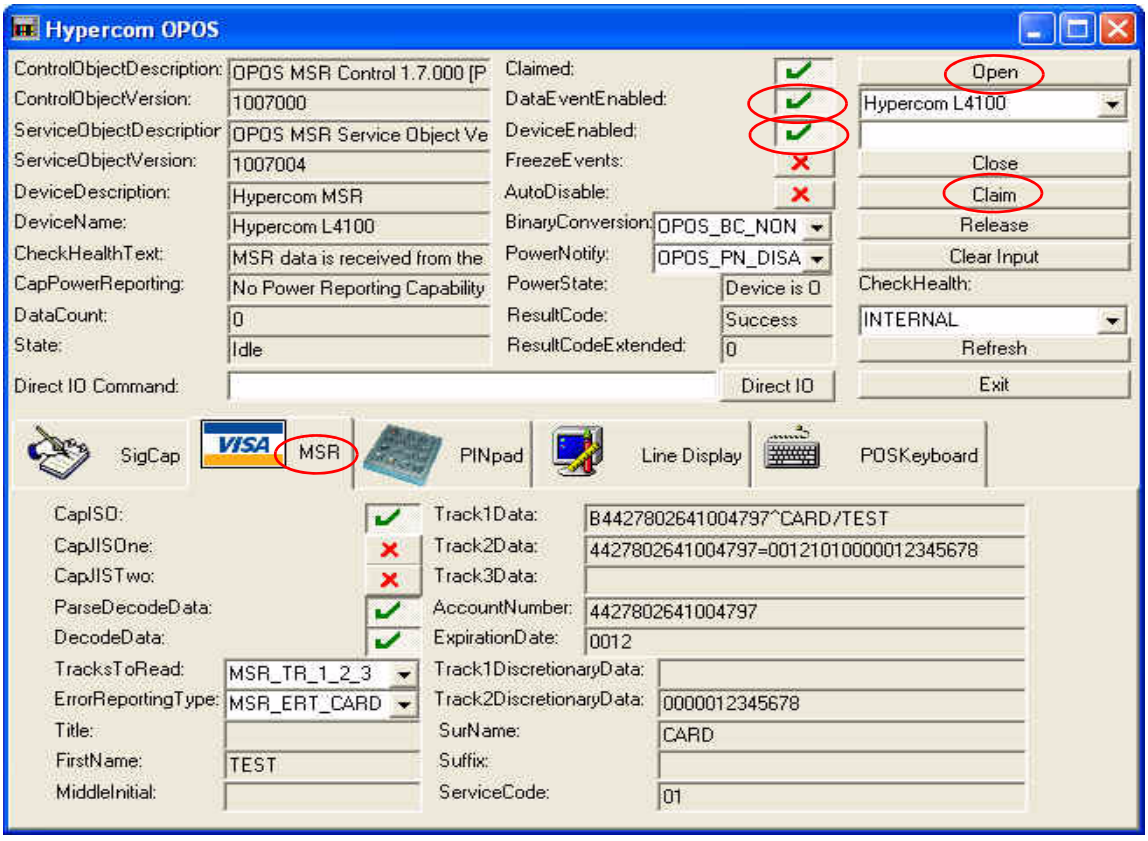

Figure 2-2 *OPOS MSR Tab*

To configure MSR:

1. From the OPOS screen, select the name of the terminal from the Terminal pull-down menu or ensure that the correct terminal is selected in the upper right corner of the screen.
2. Click the **MSR** tab.
3. Click **Open**.
4. Click **Claim**.

   NOTE When the Claim button is clicked, the **`Claimed`**: field at the top of the OPOS window changes from a red "X" to a green checkmark.

5. Click **DeviceEnabled.**

   NOTE When DeviceEnabled is selected, the red "X" on the button changes to a green checkmark.

6. Click **DataEvent Enabled**.

   NOTE When DataEventEnabled is selected, the red "X" on the button changes to a green checkmark.

7. Swipe or wave the card on the reader.
8. If you need to view attributes listed on the OPOS MSR Device Control window, see *Table 2-2* for descriptions of the properties.

Table 2-2 *MSR Property Descriptions*

Property	Description	How Initialized
AccountNumber	Holds the account number obtained from the most recently swiped card.	Initialized to the empty string if: n The field was not included in the track data obtained, or n The track data format was not one of those listed in the ParseDecodeData property description, or n ParseDecodeData is false.
CapISO	If true, the MSR device supports ISO cards.	Open method.
CapJISOne	If true, the MSR device supports JIS Type-I cards. JIS-I cards are a superset of ISO cards. Therefore, if CapJISOne is true, then it is implied that CapISO is also true.	Open method.
CapJISTwo	If true, the MSR device supports JIS Type-II cards.	Open method.

Table 2-2 *MSR Property Descriptions (Continued)*

Property	Description	How Initialized
DecodeData	If false, the Track1Data, Track2Data, and Track3Data properties are mapped from its original encoded bit sequence (as it exists on the magnetic card) to its corresponding decoded ASCII bit sequence. This conversion is mainly of relevance for data that is not of the 7-bit format, since 7-bit data needs no decoding to decipher its corresponding alphanumeric and/or Katakana characters. **Note:** Setting this property to false automatically sets ParseDecodeData to false. For more information on data decoding by card type, track, and track data format, see *Table 2-3 on page 2-13*.	Open method.

Table 2-2 *MSR Property Descriptions (Continued)*

Property	Description	How Initialized
ErrorReportingType	Holds the type of errors to report via ErrorEvents. This property has one of the following values: • MSR_ERT_CARD Report errors at a card level. • MSR_ERT_TRACK Report errors at a track level. An error is reported by an ErrorEvent when a card is swiped, and one or more of the tracks specified by the TracksToRead property contains data with errors. When the ErrorEvent is delivered to the application, two types of error reporting are supported: • Card level: A general error status is given, with no data returned. This level should be used when a simple pass/fail of the card data is sufficient. • Track level: The control can return an extended status with a separate status for each of the tracks. Also, for those tracks that contain valid data or no data, the track's properties are updated as with a DataEvent. For those tracks that contain invalid data, the track's properties are set to empty. This level should be used when the application may be able to utilize a successfully read track or tracks when another of the tracks contain errors. For example, suppose TracksToRead is MSR_1_2_3, and a swiped card contains good track 1 and 2 data, but track 3 data contains "random noise" that is flagged as an error by the MSR. With track level error reporting, the ErrorEvent sets the track 1 and 2 properties with the valid data, sets the track 3 properties to empty, and returns an error code indicating the status of each track.	Initialized to MSR_ERT_CARD by the open method.
ExpirationDate	Holds the expiration date obtained from the most recently swiped card.	Initialized to the empty string if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.

Table 2-2 *MSR Property Descriptions (Continued)*

Property	Description	How Initialized
FirstName	Holds the first name obtained from the most recently swiped card.	Initialized to the empty string if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.
MiddleInitial	Holds the middle initial obtained from the most recent swiped card.	Initialized to the empty string if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.
ParseDecodeData	When true, the decoded data contained within the Track1Data and Track2Data properties is further separated into fields for access via various other properties. Track3Data is not parsed because its data content is of an open format defined by the card user. JIS-1 Track 1 Format C and ISO Track 1 Format C data are not parsed for similar reasons. The parsed data properties are the defined possible fields for cards with data consisting of the following formats: • JIS-I / ISO Track 1 Format A • JIS-I / ISO Track 1 Format B • JIS-I / ISO Track 1 VISA Format • JIS-I / ISO Track 2 Format **Note:** Setting this property to true automatically sets DecodeData to true.	Initialized to true by the open method.
ServiceCode	Holds the service code obtained from the most recently swiped card.	Initialized to the empty string if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.

Table 2-2 *MSR Property Descriptions (Continued)*

Property	Description	How Initialized
Suffix	Holds the suffix obtained from the most recently swiped card.	Initialized to the empty string if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.
Surname	Holds the surname obtained from the most recently swiped card.	Initialized to the empty string if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.
Title	Holds the title obtained from the most recently swiped card.	Initialized to the empty string if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.
Track1Data	Holds the track 1 data obtained from the most recently swiped card. If TransmitSentinels is false, this property contains track data between but not including the start and end sentinels. If TransmitSentinels is true, then the start and end sentinels are included. If DecodeData is true, then the data returned by this property has been decoded from the “raw” format. The data may also be parsed into other properties when the ParseDecodeData property is set. A zero length array indicates that the track was not accessible.	The array length will be zero length if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.

Table 2-2 *MSR Property Descriptions (Continued)*

Property	Description	How Initialized
Track1DiscretionaryData	Holds the track 1 discretionary data obtained from the most recently swiped card. **Note:** The amount of data contained in this property varies widely depending upon the format of the track 1 data.	The array length will be zero length if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.
Track2Data	Holds the track 2 data obtained from the most recently swiped card. If TransmitSentinels is false, this property contains track data between but not including the start and end sentinels. If TransmitSentinels is true, then the start and end sentinels are included. If DecodeData is true, then the data returned by this property has been decoded from the “raw” format. The data may also be parsed into other properties when the ParseDecodeData property is set. A zero length array indicates that the track was not accessible.	The array length will be zero length if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.

Table 2-2 *MSR Property Descriptions (Continued)*

Property	Description	How Initialized
Track2DiscretionaryData	Holds the track 2 discretionary data obtained from the most recently swiped card. **Note:** The amount of data contained in this property varies widely depending upon the format of the track 2 data.	The array length will be zero length if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.
Track3Data	Holds the track 3 data obtained from the most recently swiped card. If TransmitSentinels is false, this property contains track data between but not including the start and end sentinels. If TransmitSentinels is true, then the start and end sentinels are included. If DecodeData is true, then the data returned by this property has been decoded from the “raw” format. The data may also be parsed into other properties when the ParseDecodeData property is set. A zero length array indicates that the track was not accessible.	The array length will be zero length if: • The field was not included in the track data obtained, or • The track data format was not one of those listed in the ParseDecodeData property description, or • ParseDecodeData is false.

Table 2-2 *MSR Property Descriptions (Continued)*

Property	Description	How Initialized
TracksToRead	Holds the track data that the application wishes to have placed into Track1Data, Track2Data, and Track3Data properties following a card swipe. This property has one of the following values: • MSR_TR_1 Obtain track 1. • MSR_TR_2 Obtain track 2. • MSR_TR_3 Obtain track 3. • MSR_TR_1_2 Obtain tracks 1 and 2. • MSR_TR_1_3 Obtain tracks 1 and 3. • MSR_TR_2_3 Obtain tracks 2 and 3. • MSR_TR_1_2_3 Obtain tracks 1, 2, and 3. Decreasing the required number of tracks may provide a greater swipe success rate and somewhat greater responsiveness by removing the processing for unaccessed data. TracksToRead does not indicate a capability of the MSR hardware unit. It is an application configurable property representing which track(s) will have their data obtained, potentially decoded, and returned, if possible.	Initialized to MSR_1_2_3 by the open method.

Table 2-3 lists the decoding that takes place for each card type, track, and track data format.

Table 2-3 *MSR Data Decoding*

Card Type	Track Data Property	Raw Data Format	Raw Bytes	Decoded Values
ISO	Track1Data	6-Bit	0x00 - 0x3F	0x20 through 0x5F
	Track2Data	4-Bit	0x00 - 0x0F	0x30 through 0x3F
	Track3Data	4-Bit	0x00 - 0x0F	0x30 through 0x3F

Table 2-3 *MSR Data Decoding (Continued)*

Card Type	Track Data Property	Raw Data Format	Raw Bytes	Decoded Values
JIS-I	Track1Data	6-Bit	0x00 - 0x3F	0x20 through 0x5F
	Track1Data	7-Bit	0x00 - 0x7F	Data Unaltered
	Track2Data	4-Bit	0x00 - 0x0F	0x20 through 0x3F
	Track3Data	4-Bit	0x00 - 0x0F	0x20 through 0x3F
	Track3Data	7-Bit	0x00 - 0x7F	Data Unaltered

Implementing PIN Pad

The PIN Pad Device provides a mechanism for customers to perform PIN entry. It accepts a PIN entry at its keyboard and provides an encrypted PIN to the application.

A PIN Pad performs these functions by implementing one or more PIN Pad Management Systems. Symbol supports both Master-Session (M/S) and DUKPT PIN Pad Management Systems.

The PIN Pad Management System defines the manner in which the PIN Pad performs functions such as:

- PIN encryption
- Message Authentication Code (MAC) calculation
- Key updating

The PIN Pad tab on the OPOS window allows you to open and use the PIN Pad. See the following example of an OPOS window with the PIN Pad tab selected:

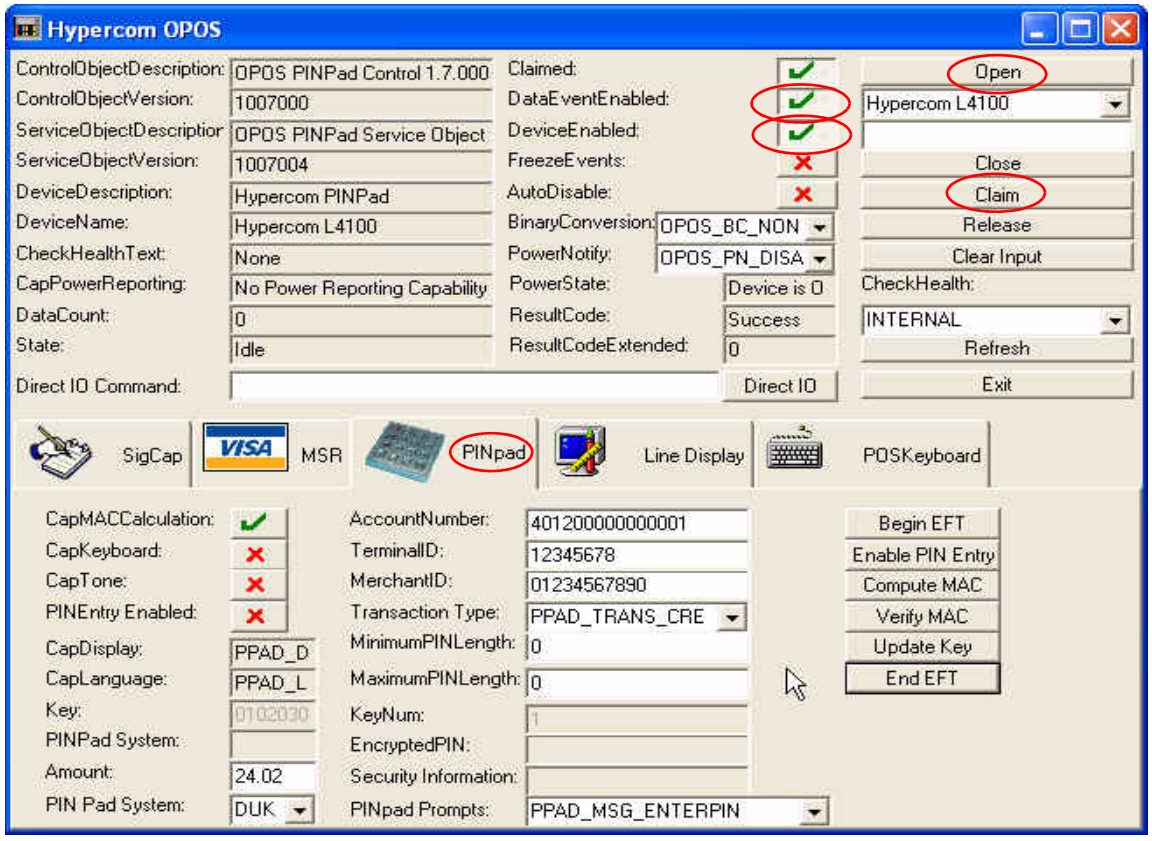

Figure 2-3 *OPOS PIN Pad Tab*

To configure the PIN Pad:

1. From the OPOS screen, select the name of the terminal from the *Terminal* pull-down menu or ensure that the correct terminal is selected in the upper right corner of the screen.
2. Click the **PINpad** tab.
3. Click **Open**.

4. Click **Claim**.

   NOTE When the Claim button is clicked, the **`Claimed`**: field at the top of the OPOS window changes from a red "X" to a green checkmark.

5. Click **DeviceEnabled**.

   NOTE When DeviceEnabled is selected, the red "X" on the button changes to a green checkmark.

6. Click **DataEvent Enabled**.

   NOTE When DataEventEnabled is selected, the red "X" on the button changes to a green checkmark.

7. To set the PIN Pad System, select either **DUKPT** or **M/S**.
8. Click **Begin EFT** to initialize the beginning of an EFT transaction.

   NOTE To access OPOS PIN Pad methods, locate the buttons in the lower right corner of the *OPOS PIN Pad Device Control* window, as shown below in *Figure 2-4* and described in *Table 2-4*.

9. Click **Enable PIN Entry** to enable PIN entry at the PIN Pad device.

   NOTE When you click **Enable Entry**, the PINEntryEnabled property changes to **`true`**.

10. If you need to view attributes listed on the OPOS PIN Pad Device Control window, see *Table 2-5* for descriptions of the properties.

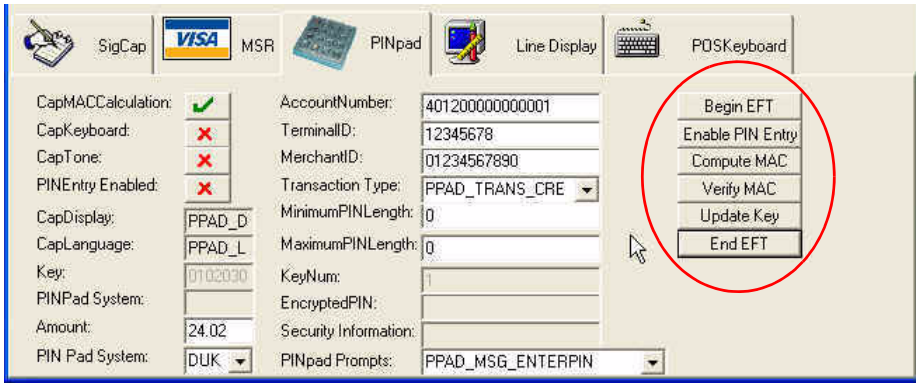

Figure 2-4 *Methods area of the OPOS PIN Pad tab*

Table 2-4 provides descriptions of the OPOS PIN Pad methods.

Table 2-4 *PIN Pad Methods Descriptions*

Method	Description	Supported and Values and Parameters
Begin EFT	Initializes the beginning of an EFT Transaction. The device will perform initialization functions, such as computing session keys. No other PIN Pad functions can be performed until this method is called.	• PINPadSystem Name of the desired PIN Pad Management System The PINPadSystem parameter has one of the following values: - M/S Master/Session - DUKPT Derived Unique Key Per Transaction • transactionHost Identifies particular EFT Transaction Host to be used for this transaction.
Compute MAC	Computes a MAC value and appends it to the designated message. Depending on the selected PIN Pad management system, the PIN Pad may also insert other fields into the message. **Note:** This method cannot be used while PIN Pad input (PIN Pad Entry) is enabled.	• inMsg The message that the application intends to send to an EFT Transaction. • outMsg Contains the result of applying the MAC calculation to inMsg. This output parameter contains a reformatted message that may actually be transmitted to an EFT Transaction Host.
Enable PIN Entry	Enables PIN Entry at the PIN Pad device. When this method is called, the PINEntryEnabled property changes to true. If the PIN Pad uses pre-defined prompts for PIN Entry, then the Prompt property changes to PPAD_MSG_ENTERPIN. When the user has completed the PIN entry operation, either by entering their PIN or by hitting Cancel, the PINEntryEnabled property changes to false. A DataEvent will be delivered to provide the encrypted PIN to the application when DataEventEnabled is set to true. **Note:** Any data entered at the PIN Pad while PINEntryEnabled is true will be supplied in encrypted form and will NOT be provided to any associated Keyboard Control Object.	N/A

Table 2-4 *PIN Pad Methods Descriptions (Continued)*

Method	Description	Supported and Values and Parameters
End EFT	Ends an EFT Transaction. The Device will perform termination functions, such as computing next transaction keys.	• PPAD_EFT_NORMAL The EFT Transaction is completed normally. Note that this does not mean that the EFT transaction was approved. It merely means that the proper sequence of messages was transmitted and received. • PPAD_EFT_ABNORMAL The proper sequence of messages was not transmitted and received.
Update Key	Provides a new encryption key to the PIN Pad. It is used only for those PIN Pad Management Systems in which new key values are sent to the terminal as a field in standard messages from the EFT Transaction Host.	• keyNum A key number. • key A Hex-ASCII value for a new key.
Verify MAC	Verify that the MAC value in a message received from an EFT Transaction Host. **Note:** This method cannot be used while PIN Entry is enabled.	• message Contains a message received from an EFT Transaction Host.

Table 2-5 *PIN Pad Property Descriptions*

Property	Description	How Initialized
AccountNumber	Holds the account number to be used for the current EFT transaction.	The application must set this property before calling the BeginEFTTransaction method.
Security Information	Holds additional security/encryption information when a DataEvent is delivered. This property is formatted as a HEX-ASCII string. The information content and internal format of this string will vary among PIN Pad Management Systems. For example, if the PIN Pad Management System is DUKPT, then this property will contain the “PIN Pad sequence number”. If the PIN Entry was cancelled, this property will contain the empty string.	Open method.
Amount	Holds the amount of the current EFT transaction. This property is a monetary value stored using an implied four decimal places. For example, an actual value of 12345 represents 1.2345.	The application must set this property before calling the beginEFTTransaction method.

Table 2-5 *PIN Pad Property Descriptions (Continued)*

Property	Description	How Initialized
PINpad Prompts	Holds a comma-separated string representation of the supported values for the Prompt property. The full set of supported Prompt values are shown below: • PPAD_MSG_ENTERPIN (1) Enter pin number on the PIN Pad. • PPAD_MSG_PLEASEWAIT (2) The system is processing. Wait. • PPAD_MSG_ENTERVALIDPIN (3) The pin that was entered is not correct. Enter the correct pin number. • PPAD_MSG_RETRIESEXCEEDED (4) The user has failed to enter the correct pin number and the maximum number of attempts has been exceeded. • PPAD_MSG_APPROVED (5) The request has been approved. • PPAD_MSG_DECLINED (6) The EFT Transaction Host has declined to perform the requested function. • PPAD_MSG_CANCELED (7) The request is cancelled. • PPAD_MSG_AMOUNTOK (8) Enter Yes/No to approve the amount. • PPAD_MSG_NOTREADY (9) PIN Pad is not ready for use. • PPAD_MSG_IDLE (10) The system is Idle. • PPAD_MSG_SLIDE_CARD (11) Slide card through the integrated MSR. • PPAD_MSG_INSERTCARD (12) Insert (smart) card. • PPAD_MSG_SELECTCARDTYPE (13) Select the card type (typically credit or debit). **Note:** Values 1000 and above are reserved for device-specific defined values.	Open method.

Table 2-5 *PIN Pad Property Descriptions (Continued)*

Property	Description	How Initialized
CapDisplay	Defines the operations that the application may perform on the PIN Pad display. The set of supported values are shown below: • PPAD_DISP_UNRESTRICTED The application can use the PIN Pad display in an unrestricted manner to display messages. In this case, an associated Line Display Control Object is the interface to the PIN Pad display. The application must call Line Display methods to manipulate the display. • PPAD_DISP_PINRESTRICTED The application can use the PIN Pad display in an unrestricted manner except during PIN Entry. The PIN Pad will display a pre-defined message during PIN entry. • PPAD_DISP_RESTRICTED_LIST The application cannot specify the text of messages to display. It can only select from a list of pre-defined messages. There is no associated Line Display Device Control. • PPAD_DISP_RESTRICTED_ORDER The application cannot specify the text of messages to display. It can only select from a list of pre-defined messages. The selections must occur in a pre-defined acceptable order. There is no associated Line Display Device Control. • PPAD_DISP_NONE The PIN Pad does not have the PIN Pad display.	Open method.
CapKeyboard	If true, the application can use the PIN Pad to obtain input. The application will use an associated POS Keyboard Device Control object as the interface to the PIN Pad keyboard. Note that the associated POS Keyboard Control is effectively disabled while PINEntryEnabled is true. If false, the application cannot obtain input directly from the PIN Pad keyboard.	Open method.

Table 2-5 *PIN Pad Property Descriptions (Continued)*

Property	Description	How Initialized
CapLangage	Defines the capabilities that the application has to select the language of pre-defined messages. This property has one of the following values: • PPAD_LANG_NONE The PIN Pad supports no pre-defined prompt messages. The property will be set to this value if CapDisplay is set to PPAD_DISP_UNRESTRICTED. • PPAD_LANG_ONE The PIN Pad supports pre-defined prompt messages in one language. • PPAD_LANG_PINRESTRICTED The PIN Pad cannot change prompt languages during PIN Entry. The application must set the desired value into the PromptLanguage property before calling enablePINEntry. • PPAD_LANG_UNRESTRICTED The application can change the language of pre-defined prompt messages at any time. The currently displayed message will change immediately.	Open method.
CapMACCalculation	If true, the PIN Pad supports MAC calculation.	Open method.
CapTone	If true, the PIN Pad has a Tone Indicator. The Tone Indicator may be accessed by use of an associated Tone Indicator Control. If false, there is no Tone Indicator.	Open method.
EncryptedPIN	Holds the value of the Encrypted PIN after a DataEvent. This property will be formatted as a hexadecimal ASCII string. Each character is in the ranges “0” through “9” or “A” through “F”. Each pair of characters is the hexadecimal representation of a byte. For example, if the first four characters are “12FA”, then the first two bytes of the PIN are 12 hexadecimal (18) and FA hexadecimal (250). If the PIN Entry was cancelled, this property will contain the empty string.	Open method.
MaximumPINLength	Holds the maximum acceptable number of digits in a PIN. This property must be set to a default value by the open method. If the application wishes to change this property, it should be set before the enablePINEntry method is called. Note that in some implementations, this value cannot be changed by the application.	Open method.
MerchantID	Holds the Merchant ID, as it is known to the EFT Transaction Host. The application must set this property before calling the beginEFTTransaction method.	Open method.

Table 2-5 *PIN Pad Property Descriptions (Continued)*

Property	Description	How Initialized
MinimumPINLength	Holds the minimum acceptable number of digits in a PIN. This property will be set to a default value by the open method. If the application wishes to change this property, it should be set before the enablePINEntry method is called. **Note:** In some implementations, this value cannot be changed by the application.	Open method.
PINEntryEnabled	If true, the PIN entry operation is enabled.	Set when the enablePINEntry method is called. Set to false when the user has completed the PINEntry operation or when the endEFTTransaction method has completed.
TerminalID	Holds the terminal ID.	The application must set this property before calling the beginEFTTransaction method.
TransactionType	Holds the type of the current EFT Transaction. The property may have one of the following values: • PPAD_TRANS_DEBIT Debit (decrease) the specified account • PPAD_TRANS_CREDIT Credit (increase) the specified account • PPAD_TRANS_INQ (Balance) Inquiry • PPAD_TRANS_RECONCILE Reconciliation/Settlement • PPAD_TRANS_ADMIN Administrative Transaction	The application must set this property before calling the beginEFTTransaction method.

Implementing Line Display

The Line Display Device supports text character display. The default mode of the display is character display output.

If enabled by the application, the Line Display Device can also support:

- Windowing with marquee-like scrolling of the window. The display may support vertical or horizontal marquees, or both.
- A waiting period between displaying characters, for a teletype effect.
- Character-level or device-level blinking at adjustable blink rates.
- Character-level or device-level reverse video.
- One or more descriptors. Descriptors are small indicators with a fixed label, and are typically used to indicate transaction states such as item, total, and change.
- Device brightness control, with one or more levels of device dimming.
- Various cursor attributes including underline, block, and reverse video.
- Glyphs which represent pixel level user definition of character cells.
- Changing screen modes (the number of rows and columns supported by the device).
- Setting and displaying bitmaps.

The Line Display tab on the OPOS window allows you to open and use the Line Display. See the following example of an OPOS window with the Line Display tab selected:

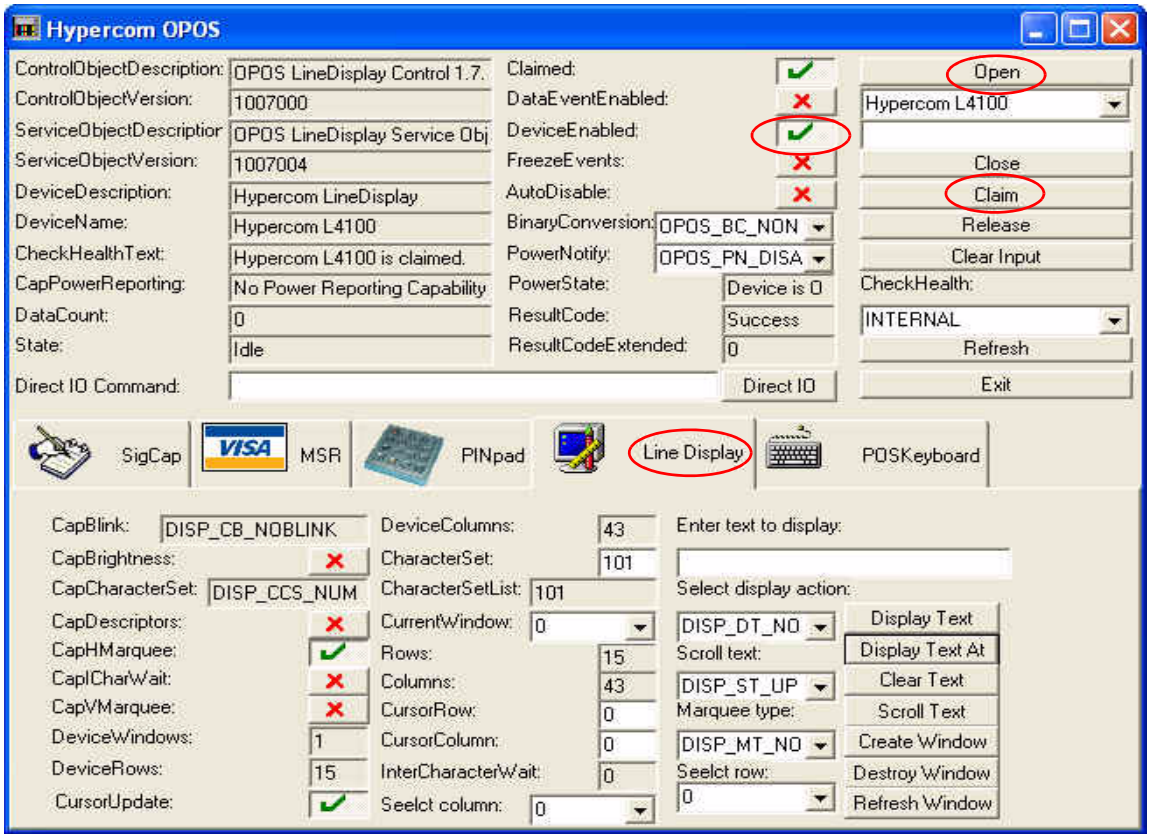

Figure 2-5 *OPOS Line Display tab*

To configure Line Display:

1. From the OPOS screen, select the name of the terminal from the *Terminal* pull-down menu or ensure that the correct terminal is selected in the upper right corner of the screen.
2. Click the **Line Display** tab.
3. Click **Open**.
4. Click **Claim**.

   NOTE When the Claim button is clicked, the **`Claimed`**: field at the top of the OPOS window changes from a red "X" to a green checkmark.

5. Select the **Device Enabled** check box.

   NOTE When DeviceEnabled is selected, the red "X" on the button changes to a green checkmark.

6. To access OPOS Line Display methods, locate the buttons in the lower right corner of the OPOS Line Display Device Control window, as shown below in *Figure 2-6* and described in *Table 2-6*.

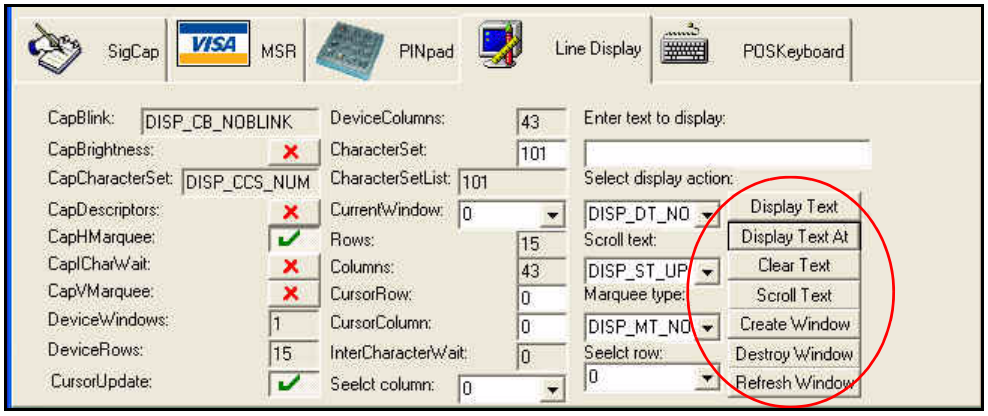

Figure 2-6 *Methods Area of the OPOS Line Display Tab*

7. If you need to view attributes listed on the OPOS POS Keyboard Device Control window, see *Table 2-7 on page 2-29* for descriptions of the properties.

Table 2-6 provides descriptions of the OPOS Line Display methods.

Table 2-6 *Line Display Methods Descriptions*

Method	Description	Supported Values
Clear Text	Clears the current window to blanks, sets CursorRow and CursorColumn to zero, and re synchronizes the beginning of the window with the start of the viewport. All clears all bitmaps displayed in the window.	• Immediate Mode or Teletype Mode The viewport is also cleared immediately. • Marquee Init Mode The viewport is not changed. • Marquee On Mode This method is not valid.

Table 2-6 *Line Display Methods Descriptions (Continued)*

Method	Description	Supported Values
Create Window	Creates a viewport over the portion of the display given by the first four parameters. The window size is given by the last two parameters. Valid window row values range from zero to one less than windowHeight and column values range from zero to one less than windowWidth. The window size must be at least as large as the viewport size. The window size may be larger than the viewport size in one direction. Using the window marquee properties MarqueeType, MarqueeFormat, MarqueeWindowWait, and MarqueeRepeat Wait, such a window may be continuously scrolled in a marquee fashion. When successful, createWindow sets the CurrentWindow property to the window number assigned to this window.	The following properties are maintained for each window, and are initialized as given: • Rows Set to windowHeight. • Columns Set to windowWidth • CursorRow Set to 0. • CursorColumn Set to 0. • CursorType Set to DISP_CT_NONE (or the appropriate cursor type if CapCursorType has DISP_CCT_FIXED set). • CursorUpdate Set to true. • MarqueeType Set to DISP_MT_NONE • MarqueeFormat Set to DISP_MF_WALK • MarqueeUnitWait Set to 0. • MarqueeRepeatWait Set to 0. • InterCharacterWait Set to 0.
Destroy Window	Destroys the current window. The characters displayed in its viewport are not changed. CurrentWindow is set to window 0. The device window and the associated window properties are updated.	N/A

Table 2-6 *Line Display Methods Descriptions (Continued)*

Method	Description	Supported Values
Display Text	The characters in data are processed beginning at the location specified by CursorRow and CursorColumn, and continue in succeeding character positions. Any previous data in a character position is overwritten, including character and bitmap data. Character processing continues to the next row when the end of a window row is reached. If the end of the window is reached with additional characters to be processed, then the window is scrolled upward by one row and the bottom row is set to blanks. If CursorUpdate is true, then CursorRow and CursorColumn are updated to point to the character position following the last character of data. **Note:** Scrolling will not occur when the last character of data is placed at the end of a row. In this case, when CursorUpdate is true, then CursorRow is set to the row containing the last character, and CursorColumn is set to Columns (that is, to one more than the final character of the row.) This stipulation ensures that the display does not scroll when a character is written into its last position. Instead, the Service will wait until another character is written before scrolling the window.	• data The string of characters to display. • attribute The display attribute for the text. Must be either: - DISP_DT_NORMAL - DISP_DT_BLINK
Display Text At	The characters in data are processed beginning at the window location specified by the row and column parameters, and continuing in succeeding columns. The operational characteristics of the displayTextAt method are the same as the displayText method. This method has the same effect as setting the CursorRow to row, setting CursorColumn to column, and calling the displayText method.	• row The start row for the text. • column The start column for the text • data The string of characters to display • attribute The display attribute for the text. Must be either: - DISP_DT_NORMAL - DISP_DT_BLINK
Refresh Window	Changes the current window to window, then re-displays its viewport. Neither the mapping of the window to its viewport nor the window's cursor position is changed. This function may be used to restore a window after another window has overwritten some of its viewport.	N/A

Table 2-6 *Line Display Methods Descriptions (Continued)*

Method	Description	Supported Values
Scroll Text	Scrolls the current window. This function is valid only in Immediate Mode. If the window size for the scroll direction matches its viewport size, then the window data is scrolled, the last units row or columns are set to spaces, and the viewport is updated. If the window contains bitmap data, it is also scrolled. If the window size for the scroll direction is larger than its viewport, then the window data is not changed. Instead the mapping of the window into the viewport is moved the specified direction. The window data is not altered, but the viewport is updated. If scrolling by units would go beyond the beginning of the window data, then the window is scrolled so that the first viewport row or column contains the first window row or column. If scrolling by units would go beyond the end of the window data, then the window is scrolled so that the last viewport row or column contains the last window row or column.	The direction parameter indicates the scrolling direction, and is one of the following values: • DISP_ST_UP Scroll the window up. • DISP_ST_DOWN Scroll the window down. • DISP_ST_LEFT Scroll the window left. • DISP_ST_RIGHT Scroll the window right.

Table 2-7 *Line Display Property Descriptions*

Property	Description	How Initialized/Updated
CapBlink	Holds the character blink capability of the device. It has one of the following values: • DISP_CB_NOBLINK Blinking is not valid. Value is 0. • DISP_CB_BLINKALL Blinking is supported. The entire contents of the display are either blinking or in a steady state. • DISP_CB_BLINKEACH Blinking is supported. Each character may be individually set to blink or be in a steady state.	Open method.
CapBrightness	If true, then the brightness control is supported.	Open method.
CapCharacterSet	Holds the default character set capability. It has one of the following values: • DISP_CCS_NUMERIC The default character set supports numeric data, plus space, minus, and period. • DISP_CCS_ALPHA The default character set supports uppercase alphabetic plus numeric, space, minus, and period. • DISP_CCS_ASCII The default character set supports all ASCII character 0x20 through 0x7F. • DISP_CCS_KANA The default character set supports partial code page 932, including ASCII characters 0x20 through 0x7F and the Japanese Kana characters 0xA1 through 0xDF, but excluding the Japanese Kanji characters. • DISP_CCS_KANJI The default character set supports code page 932, including the Shift-JIS Kanji characters, Levels 1 and 2. • DISP_CCS_UNICODE The default character set supports UNICODE. **Note:** The default character set may contain a superset of these ranges. The initial CharacterSet property may be examined for additional information.	Open method.
CapDescriptors	If true, then the display supports descriptors.	Open method.
CapHMarquee	If true, the display supports horizontal marquee windows.	Open method.
CapICharWait	If true, the display supports intercharacter wait.	Open method.
CapVMarquee	If true, the display supports vertical marquee windows.	Open method.

Table 2-7 *Line Display Property Descriptions (Continued)*

Property	Description	How Initialized/Updated
CharacterSet	Holds the character set for displaying characters. It has one of the following values: • Range 101 - 199 Device-specific character sets that do not match a code page or the ASCII or ANSI character sets. • Range 400 - 990 Code page. Matches one of the standard values. • DISP_CS_UNICODE The character set supports UNICODE. The value of this constant is 997. • DISP_CS_ASCII The ASCII character set, supporting the ASCII characters 0x20 through 0x7F. The value of this constant is 998. • DISP_CS_ANSI The ANSI character set. The value of this constant is 999. **Note:** This value is guaranteed to support at least the set of characters specified by CapCharacterSet.	Initialized to an appropriate value when the device is first enabled following the open method.
CharacterSetList	Holds the character set numbers supported. It consists of ASCII numeric set numbers separated by commas. For example, if the string is “101,850,999”, then the device supports a device-specific character set, code page 850, and the ANSI character set.	Open method.
Columns	Holds the number of columns for this window. For window 0, this property is the same as DeviceColumns. For other windows, it may be less than or greater than DeviceColumns.	Initialized to DeviceColumns by the open method. Updated when CurrentWindow is set and when createWindow or destroyWindow are called.
CurrentWindow	Holds the current window to which text is displayed. Several properties are associated with each window: Rows, Columns, CursorRow, CursorColumn, CursorUpdate, CursorType, MarqueeFormat, MarqueeType, MarqueeUnitWait, MarqueeRepeatWait, and InterCharacterWait. When set, this property changes the current window and sets the associated properties to their values for this window. Setting a window does not refresh its viewport. If this window and another window’s viewport overlap, and the other window has changed the viewport, then refreshWindow may be called to restore this window’s viewport contents.	Initialized to zero by the open method. Updated when createWindow or destroyWindow are called.

Table 2-7 *Line Display Property Descriptions (Continued)*

Property	Description	How Initialized/Updated
CursorColumn	Holds the column in the current window to which the next displayed character will be output. Legal values range from zero through Columns. (See displayText for a note on the interpretation of CursorColumn = Columns.)	Initialized to zero by the open and createWindow methods. Updated when CurrentWindow is set or clearText, displayTextAt, or destroyWindow is called. It is also updated when displayText is called if CursorUpdate is true.
CursorRow	Holds the row in the current window to which the next displayed character will be output. Legal values range from zero through one less than Rows.	Initialized by the open and createWindows methods. Updated when CurrentWindow is set or clearText, displayTextAt, or destroyWindow is called. It is also updated when displayText is called if CursorUpdate is true.
CursorUpdate	When true, CursorRow and CursorColumn will be updated to point to the character beyond the last character output when characters are displayed using the displayText or displayTextAt method. When false, the cursor properties will not be updated when characters are displayed. Note: This property is maintained for each window.	Initialized to true by the open and createWindow methods. Updated when CurrentWindow is set or destroyWindow is called.
DeviceColumns	Holds the number of columns on this device.	Initialized by the open method. Updated when the ScreenMode property is changed.
DeviceRows	Holds the number of rows on this device	Initialized by the open method. Updated when the ScreenMode property is changed.
DeviceWindows	Holds the maximum window number supported by this device. A value of zero indicates that only the device window is supported and that no windows may be created.	Open method.

Table 2-7 *Line Display Property Descriptions (Continued)*

Property	Description	How Initialized/Updated
InterCharacterWait	Holds the wait time between displaying each character with the displayText and displayTextAt methods. This provides a “teletype” appearance when displaying text. This property is only used if the window is not in Marquee Mode, that is, MarqueeType must be DISP_MT_NONE. When non-zero and the window is not in Marquee Mode, the window is in Teletype Mode: displayText and displayTextAt requests are enqueued and processed in the order they are received. This property specifies the time to wait between outputting each character into the viewport. The wait time is the specified number of milliseconds. (Note that the system timer resolution may reduce the precision of the wait time.) If CursorUpdate is true, CursorRow and CursorColumn are updated to their final values before displayText or displayTextAt returns, even though all of its data may not yet be displayed. When this property is zero and the window is not in Marquee Mode, Immediate Mode is in effect, so that characters are processed as quickly as possible. If capICharWait is false, then inter character waiting is not valid, and the value of this property is not used.	Initialized to zero by the open and createWindow methods. Updated when CurrentWindow is set or destroyWindow is called.

Table 2-7 *Line Display Property Descriptions (Continued)*

Property	Description	How Initialized/Updated
MarqueeType	Holds the marquee type for the current window. When not DISP_MT_NONE, the window is in Marquee Mode. This property has one of the following values: • DISP_MT_NONE Marquees are disabled for this window. • DISP_MT_INIT Marquee Init Mode. Changes to the window are not reflected in the viewport until this property is changed to another value. • DISP_MT_UP Scroll the window up. Not valid unless Rows is greater than the viewportHeight parameter used for the window's createWindow call, and CapVMarquee is true. • DISP_MT_DOWN Scroll the window down. Not valid unless Rows is greater than the viewportHeight parameter used for the window's createWindow call, and CapVMarquee is true. • DISP_MT_LEFT Scroll the window left. Not valid unless Columns is greater than the viewportWidth parameter used for the window's createWindow call, and CapHMarquee is true. • DISP_MT_RIGHT Scroll the window right. Not valid unless Columns is greater than the viewportWidth parameter used for the window's createWindow call, and CapHMarquee is true. A marquee is typically initialized after entering Marquee Init Mode by setting this property to DISP_MT_INIT, then calling clearText and displayText(At) methods. Then, when this property is changed to an "on" value, Marquee On Mode is entered, and the marquee begins to be displayed in the viewport beginning at the start of the window (or end if the type is right or down). When the mode is changed from Marquee On Mode to Marqee Off Mode, the marquee stops in place. A subsequent transition back to Marquee On Mode continues from its current position. When the mode is changed from Marquee On Mode to Marquee Init Mode, the marquee stops. Changes may be made to the window, then the window may be returned to Marquee On Mode to restart the marquee with the new data. **Note:** This property is always DISP_MT_NONE for window 0, the device window.	Initialized to DISP_MT_NONE by the open and createWindow methods. Updated when CurrentWindow is set or destroyWindow is called.

Table 2-7 *Line Display Property Descriptions (Continued)*

Property	Description	How Initialized/Updated
Rows	Holds the number of rows for this window. For window 0, this property is the same as DeviceRows. For other windows, it may be less or greater than DeviceRows.	Initialized to DeviceRows by the open method. Updated when CurrentWindow is set or createWindow or destroyWindow are called.

Implementing POS Keyboard

The POS Keyboard Device reads keys from a POS keyboard. A POS keyboard may be an auxiliary keyboard or it may be a virtual keyboard consisting of some or all of the keys on the system keyboard.

The POS Keyboard tab on the OPOS window allows you to open and use the POS Keyboard. See the following example of an OPOS window with the POS Keyboard tab selected:

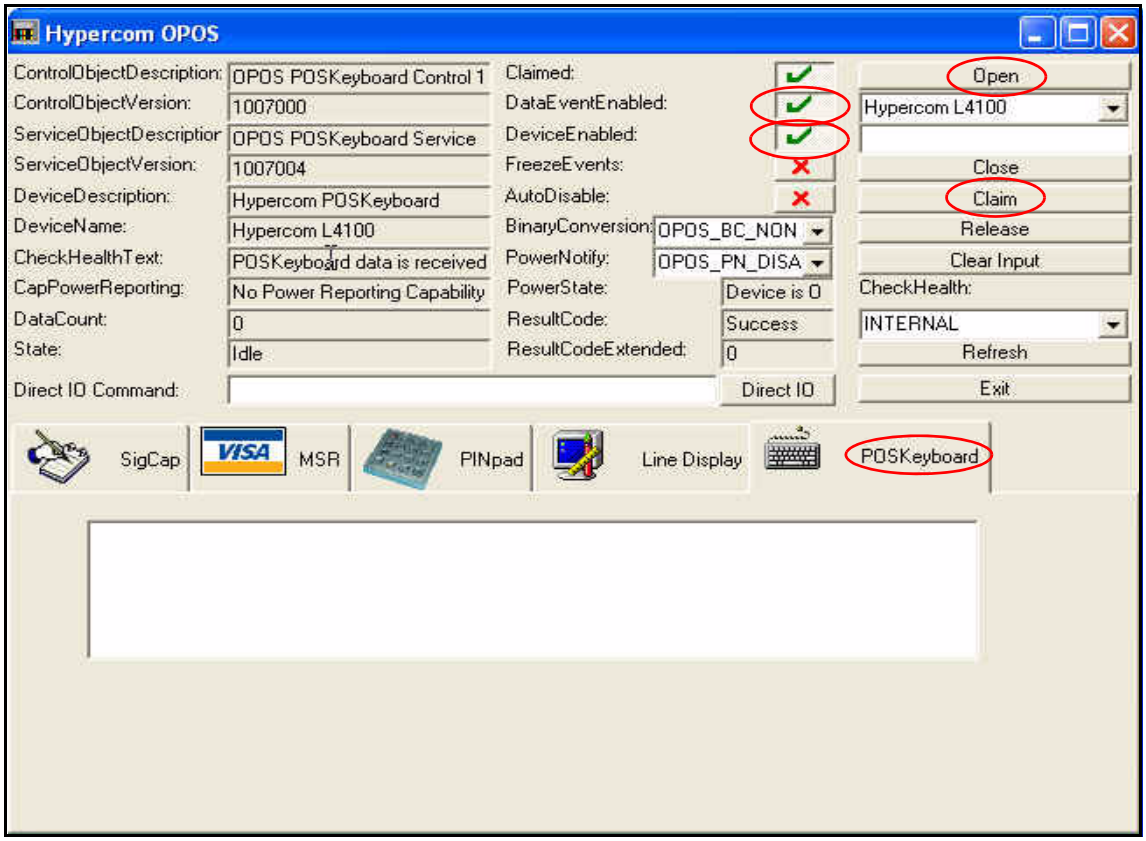

Figure 2-7 *OPOS POS Keyboard Tab*

To configure the POS Keyboard:

1. From the OPOS screen, select the name of the terminal from the Terminal pull-down menu or ensure that the correct terminal is selected in the upper right corner of the screen.
2. Click the **POS Keyboard** tab.
3. Click **Open**.
4. Click **Claim**.

   NOTE When the Claim button is clicked, the **`Claimed`**: field at the top of the OPOS window changes from a red "X" to a green checkmark.

5. Click **DeviceEnabled**.

   NOTE When DeviceEnabled is selected, the red "X" on the button changes to a green checkmark.

6. Click **DataEvent Enabled**.

   NOTE When DataEventEnabled is selected, the red "X" on the button changes to a green checkmark.

Chapter 3

Adding Direct IO Service Information

Introduction

Adding direct IO service information provides a means for Devices to provide events to the application that are not otherwise supported by Device Control. Use of this event may restrict the application program from being used with other devices which may not have any knowledge of the Service's need for this event.

To provide service information directly to the application:

1. From the selected Device Control window, click **Open**.
2. Click **Claim**.
3. Click **DeviceEnabled**.
4. Click **DataEvent Enabled**.
5. Enter the service information into the Direct IO box. See *Figure 3-1* for the location of the Direct IO settings window.

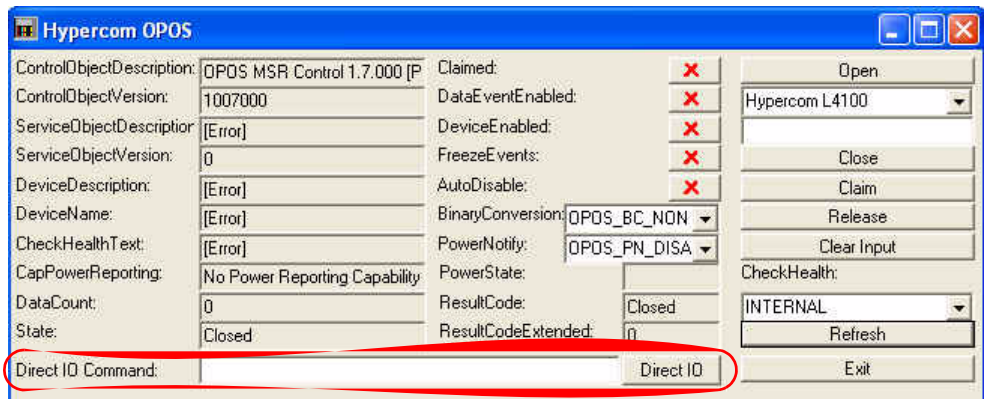

Figure 3-1 *Direct IO Settings*

6. Click **Direct IO**.

NOTE This event provides a means for the Service to provide events to the application that are not otherwise supported by the Device Control. Use of this event may restrict the application program from being used with other devices which may not have any knowledge of the Service's need for this event.

This event contains the attributes shown in *Table 3-1* below.

Table 3-1 *Direct IO Event Attributes*

Attribute	Type	Description
EventNumber	int32	Event number whose specific values are assigned by the Service.
Data	int32	Additional numeric data. Specific values vary by the EventNumber and the Service. This property is settable.
Obj	object	Additional data whose usage varies by the EventNumber and Service. This property is settable.

Chapter 4

Using Event Viewer

Introduction

The purpose of the Event Viewer feature is to log internal messages to a common repository. Messages are logged to the Windows Event Viewer application log. Internal messages include errors, warnings, information, and debugging messages. The user has the ability to filter the messages that are written to the Event Viewer by modifying the logging level in the windows registry. The Event Viewer provides the user the ability to export events, using filters, to a file.

Starting Event Viewer

To start Event Viewer:

1. Go to **Control Panel** > **Administrative Tools** > **Event Viewer**. A window similar to the following is displayed:

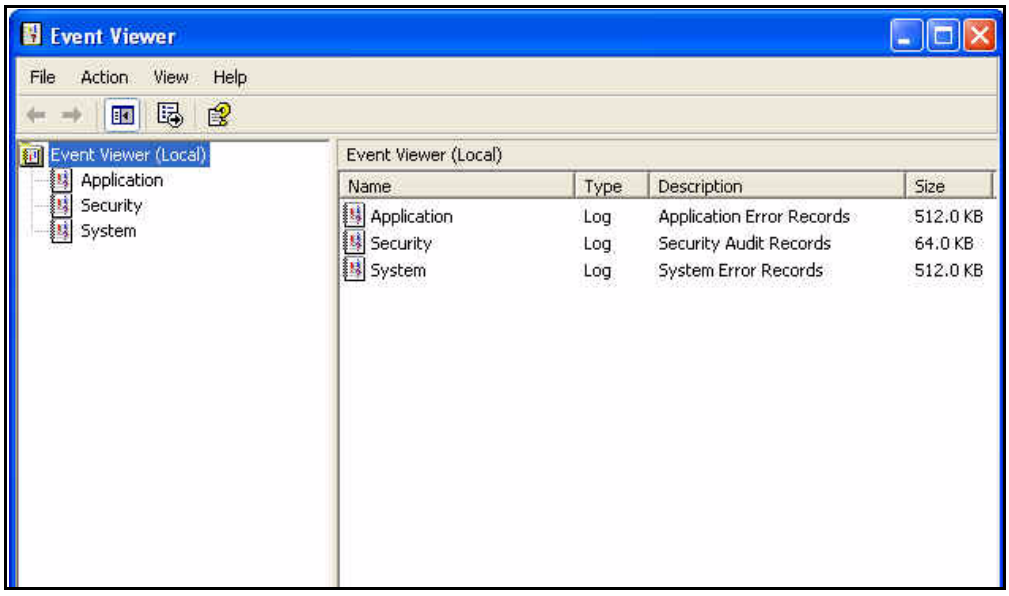

Figure 4-1 *Example of initial Event Viewer window*

2. Select **Application**. A window similar to the following is displayed:

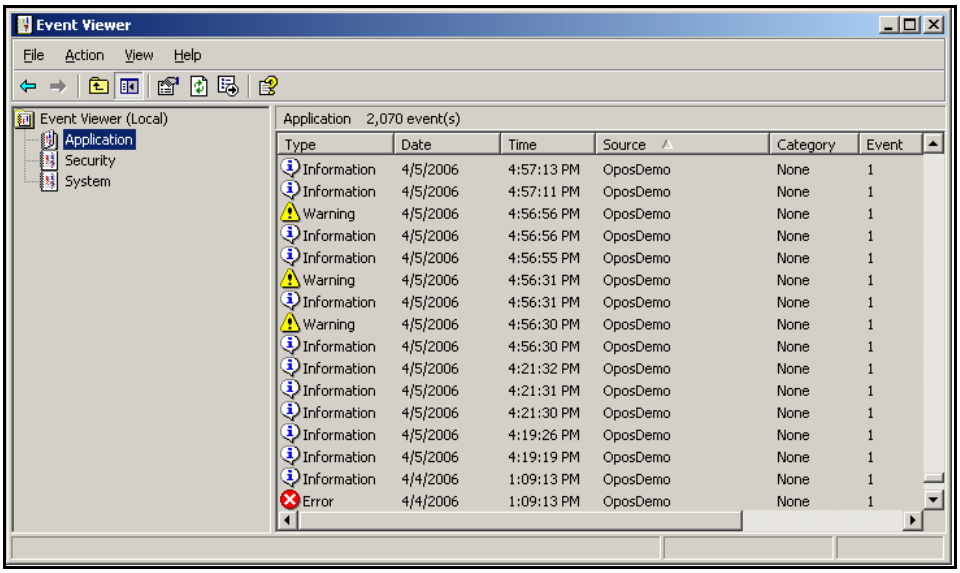
Event Viewer
File Action View Help
Event Viewer (Local)
Application
Security
System
Application 2,070 event(s)
Type Date Time Source Category Event
Information 4/5/2006 4:57:13 PM OposDemo None 1
Information 4/5/2006 4:57:11 PM OposDemo None 1
Warning 4/5/2006 4:56:56 PM OposDemo None 1
Information 4/5/2006 4:56:56 PM OposDemo None 1
Information 4/5/2006 4:56:55 PM OposDemo None 1
Warning 4/5/2006 4:56:31 PM OposDemo None 1
Information 4/5/2006 4:56:31 PM OposDemo None 1
Warning 4/5/2006 4:56:30 PM OposDemo None 1
Information 4/5/2006 4:56:30 PM OposDemo None 1
Information 4/5/2006 4:21:32 PM OposDemo None 1
Information 4/5/2006 4:21:31 PM OposDemo None 1
Information 4/5/2006 4:21:30 PM OposDemo None 1
Information 4/5/2006 4:19:26 PM OposDemo None 1
Information 4/5/2006 4:19:19 PM OposDemo None 1
Information 4/4/2006 1:09:13 PM OposDemo None 1
Error 4/4/2006 1:09:13 PM OposDemo None 1

Figure 4-2 *Example of Event Viewer entries*

Event Information

The Event Viewer 'Type' column can contain either Information or Debug messages. The 'Source' column contains the name of the ECR application. See *Figure 4-2* for examples of Event Viewer entries, including an error, warnings, and information events. For more event detail, double-click on an event Type to open an Event Properties tab as shown in *Figure 4-3*.

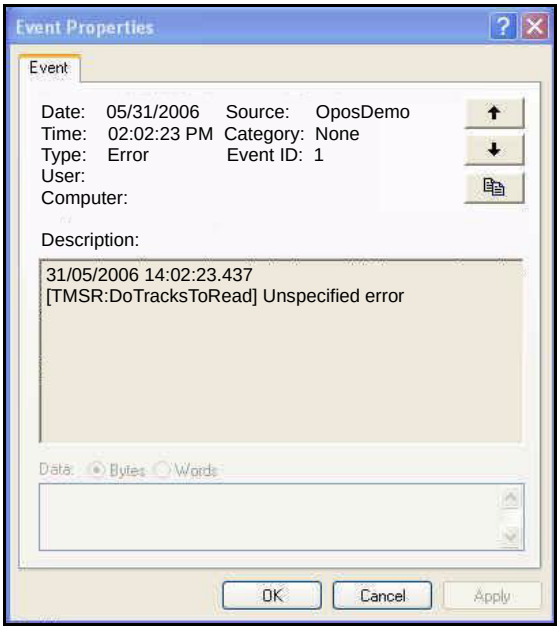

Figure 4-3 *Example of Event Properties tab*

Examples of Severity Levels

Examples of how the severity level corresponds to logged messages is provided in *Table 4-1* below.

Table 4-1 *Severity Levels and Events*

Severity Level	Event
Error	Terminal is powered off
Warning	Received PIN Block, but PINPadSO is disabled
Information	OPOS Status OK message
Debug	Tracing internal functions

Configuring Logging Levels

Logging levels are inclusive of all levels below the specified level. Each level corresponds to a specific severity of the log message. Generally, the purpose of logging events to the Event Viewer is to provide the application designer a method to debug their application by providing specific problem descriptions should an error or bug be discovered in their implementation.

✓ ***NOTE*** Initially, only the OPOS service objects will log messages to the Event Viewer. In the future, the FPE Interface, Serial, and Ethernet drivers will also log messages in a similar fashion.

See *Table 4-2* for descriptions of the logging levels:

Table 4-2 *Logging Levels*

Level	Value	Meaning
None	0	Logging is disabled
Error	1	Log only errors Note: This is the default log level.
Warning	2	Log warnings and errors
Information	4	Log info, warnings, and errors
Debug	5	Log all messages

Registry Key Name

The following registry key name is used by the Event Logger where the default '`LogLevel`' = `1`:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Hypercom\EventLog\LogLevel
```

Symbol Technologies, Inc.
One Symbol Plaza
Holtsville, New York 11742-1300
http://www.symbol.com

72E-86287-01
Revision A - November 2006